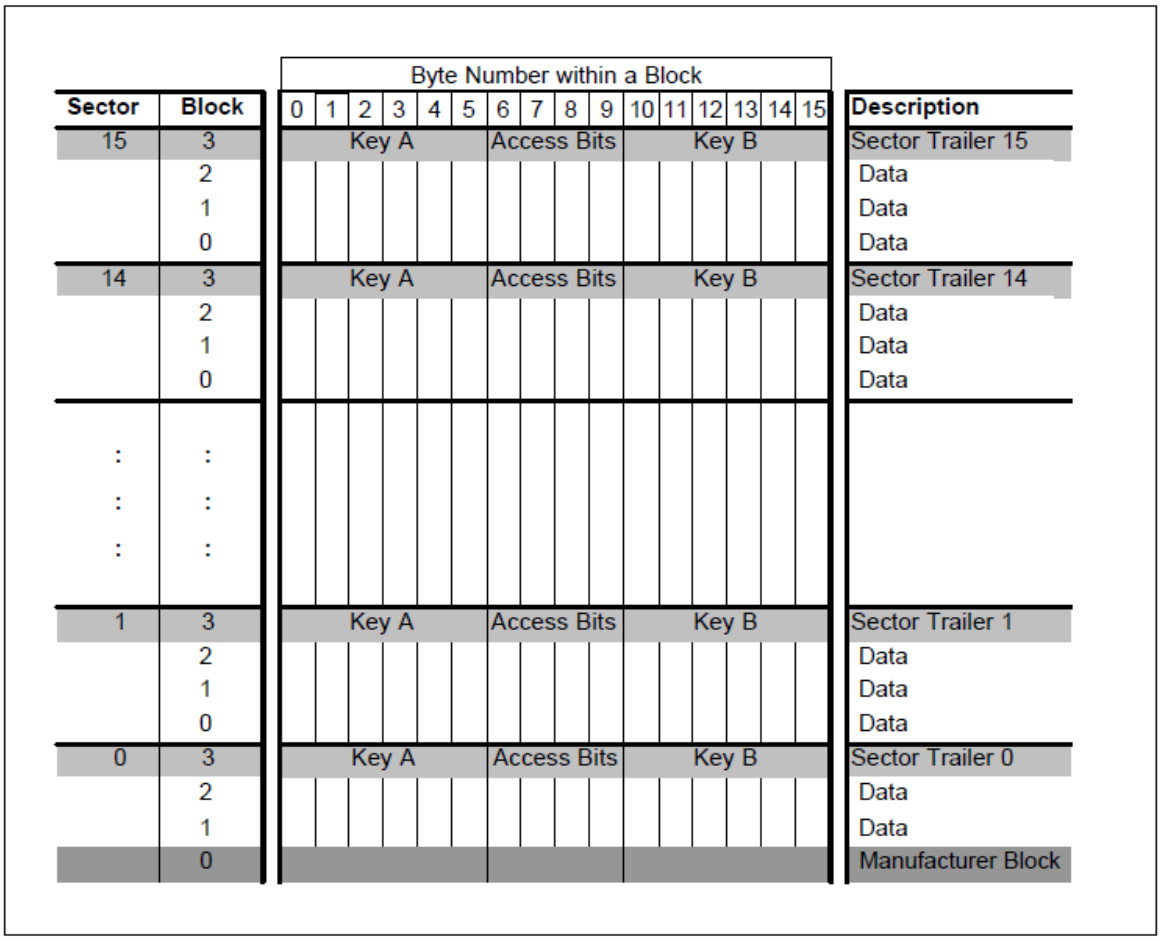
ER301 Application Programming Interface

1. S50 memory



Sector	Block	Byte Number within a Block 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15	Description
15	3	Key A | Access Bits | Key B	Sector Trailer 15
	2		Data
	1		Data
	0		Data
14	3	Key A | Access Bits | Key B	Sector Trailer 14
	2		Data
	1		Data
	0		Data
:	:		
:	:		
:	:		
1	3	Key A | Access Bits | Key B	Sector Trailer 1
	2		Data
	1		Data
	0		Data
0	3	Key A | Access Bits | Key B	Sector Trailer 0
	2		Data
	1		Data
	0		Manufacturer Block

Mifare S50 has 1k bytes, it has 16 sectors, from sector 0 to 15.
Each sector has 4 blocks, the tailor block save keys. So there are 64 blocks, the absolute address is from 0 to 63.
The key block number is x=s*4+3, s: sector number(0-15).
For more detail please see the file “Mifare_S50_en.pdf”.

2. API description

The API include two DLL files:
1.MasterRd.dll: all the API functions are included in this dll file.
2.MasterCom.dll：Serial communication file, it will be called by MasterRd.dll
This two files must be included into the program folder.
Note:

[IN]: Input
[OUT]: Output
icdev=0 (default)

3. API INFORMATION (icdev=0)

3.1 SYSTEM FUNCTION

3.1.1 INT WINAPI LIB_VER

Function: Get DLL Version
Prototype: int WINAPI lib_ver (unsigned int *pVer)
Parameter: pVer: [OUT] DLL version
Return: return 0 if successful

3.1.2 INT WINAPI RF_INIT_COM

Function: Connect
Prototype: int WINAPI rf_init_com (int port, long baud)
Parameter: port: [IN] serial port number
baud: [IN] communication baud rate, 115200 (default)
Return: return 0 if successful

3.1.3 INT WINAPI RF_CLOSEPORT

Function: Disconnect
Prototyp: int WINAPI rf_ClosePort(void)
Return: return 0 if successful

3.1.4 INT WINAPI RF_GET_MODEL

Function: Get Device Type
Prototype: int WINAPI rf_get_model (unsigned short icdev,
unsigned char *pVersion,
unsigned char *pLen)
Parameter: icdev: [IN] Device ID
pVersion: [OUT] response information
pLen: [OUT] length of response information
Return: return 0 if successful

3.1.5 INT WINAPI RF_INIT_TYPE (RFU)

Function: Set Reader contactless working mode
Prototype: int WINAPI rf_init_type(unsigned short icdev, unsigned char type)

Parameter: icdev: [IN] Device ID
type: [IN] reader working mode
Return: return 0 if successful
Explanation: this function is not effective to the readers only support single protocol.
type = 'A': set YHY632 into ISO14443A mode
type = 'B': set ISO14443B mode
type = 'r': set AT88RF020 card mode
type = '1': set ISO15693 mode
Note: this function is not used for ER301

3.1.6 INT WINAPI RF_ANTENNA_STA

Function: Manage RF Transmittal
Prototype: int WINAPI rf_antenna_sta (unsigned short icdev, unsigned char model)
Parameter: icdev: [IN] Device ID
model: [IN] transmittal state
Return: return 0 if successful
Explanation:model = 0: turn off RF antenna
model = 1: turn on RF antenna

3.1.7 INT WINAPI RF_LIGHT

Function: Manage LED
Prototype: int WINAPI rf_light (unsigned short icdev, unsigned char color)
Parameter: icdev: [IN] Device ID
color: [IN] 0 = off
1 = blue
2 = red

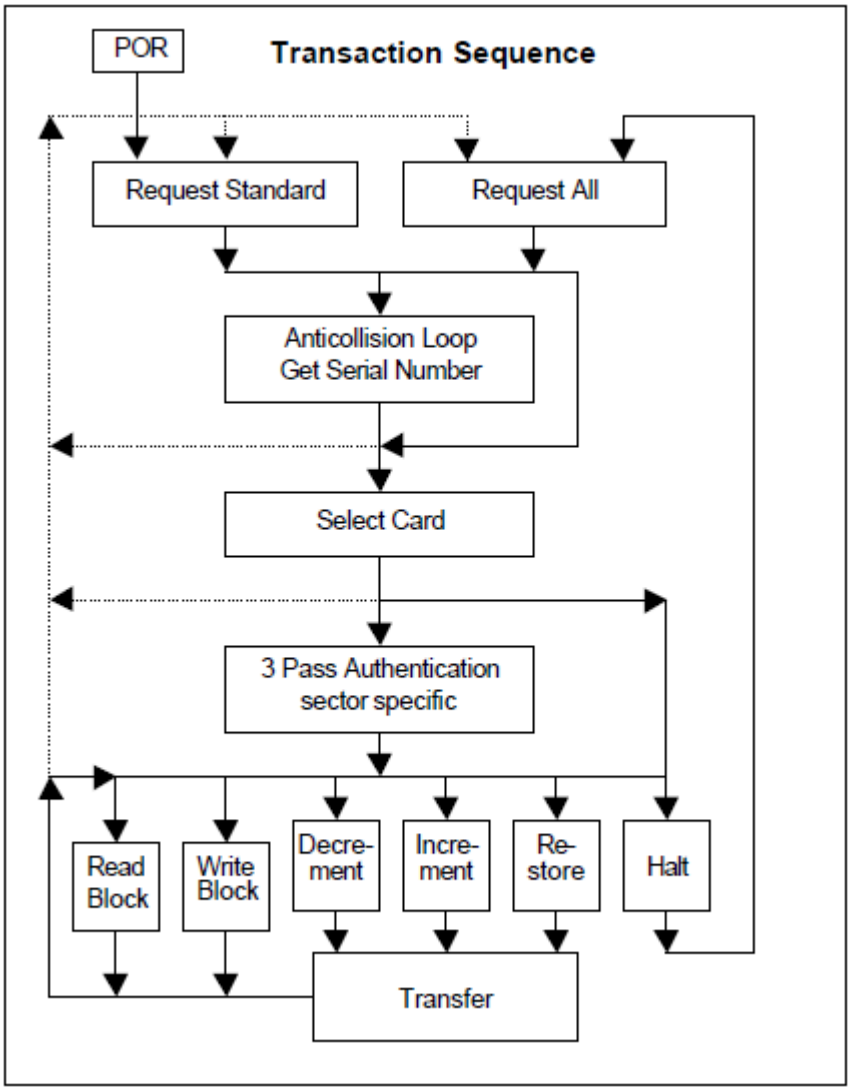
Return: return 0 if successful

3.1.8 INT WINAPI RF_BEEP

Function: beep
Prototype: int WINAPI rf_beep (unsigned short icdev, unsigned char msec)
Parameter: icdev: [IN] Device ID
msec: [IN] beep time, unit 10 Msec
Return: return 0 if successful

3.2 .1 ISO14443A FUNCTION

3.2.2 Mifare_Std



3.2.2.1 INT WINAPI RF_REQUEST

Function: ReqA
Prototype: int WINAPI rf_request (unsigned short icdev,
unsigned char model,
unsigned short *pTagType)
Parameter: icdev: [IN] Device ID
model: [IN] REQ MODE
pTagType: [OUT] response data, chip type code
Return: return 0 if successful
Explanation: mode = 0x26: REQ_STD (if the card was halt, it will not active)
mode = 0x52: REQ_ALL (can active the card even the card was halt)

3.2.2.2 INT WINAPI RF_ANTICOLL

Function: Mifare card Anticollision
Prototype: int WINAPI rf_anticoll (unsigned short icdev,
unsigned char bcnt,
unsigned char *pSnr,
unsigned char *pLen)
Parameter: icdev: [IN] Device ID
bcnt: [IN] must be 4
pSnr: [OUT] response data from card, unique serial number
pLen: [OUT] length of response data
Return: return 0 if successful

3.2.2.3 INT WINAPI RF_SELECT

Function: Mifare card Selectting
Prototype: int WINAPI rf_select (unsigned short icdev,
unsigned char *pSnr,
unsigned char snrLen,
unsigned char *pSize)
Parameter: icdev: [IN] Device ID
pSnr: [IN] card unique serial number
snrLen: [IN] length of pSnr
pSize: [OUT] response data from card, capacity code
Return: return 0 if successful
Explanation: card will be on active estate after received this command, only one TYPE_A card on active estate at the same influence range at same time.

3.2.2.4 INT WINAPI RF_M1_AUTHENTICATION2

Function: Mifare_Std Authentify
Prototype: int WINAPI rf_M1_authentication2 (unsigned short icdev,
unsigned char model,
unsigned char block,
unsigned char *pKey)
Parameter: icdev: [IN] Device ID
model: [IN] key validate mode
block: [IN] block absolute address
pKey: [IN] 6 bytes password
Return: return 0 if successful
Explanation:model = 0x60: via KeyA
model = 0x61: via KeyB

3.2.2.5 INT WINAPI RF_M1_READ

Function: MifareOne Read
Prototype: int WINAPI rf_M1_read (unsigned short icdev,

unsigned char block,
unsigned char *pData,
unsigned char *pLen)

Parameter: icdev: [IN] Device ID
block: [IN] block absolute address
pData: [OUT] response data from card
pLen: [OUT] length of response data

Return: return 0 if successful

Note: All the data store in the card is hexdecimal, for example, "RFID" store in the card is "52464944"

3.2.2.6 INT WINAPI RF_M1_WRITE

Function: Mifare_Std Write

Prototype: int WINAPI rf_M1_write (unsigned short icdev,
unsigned char block,
unsigned char *pData)

Parameter: icdev: [IN] Device ID
block: [IN] block absolute address
pData: [IN] written data, 16 bytes

Return: return 0 if successful

If you only want to write 5 bytes data,for example"12345",then you have to change it into hexdecimal "3132333435",then add 11 bytes "00" behind it, because he block needs 16 bytes data, finally the string write into the card is "31323334350000000000000000000000".

Note: use this function, the user can change the keys.

For example, you want to change the keyA of sector 02 from ffffffffffff to 313233343536 then begin write this 16 bytes (hex) into block 2*4+3=11

313233343536078069ffffffffffff

If writing successful, then you have to use new key 313233343536 to auth this sector when you read or write it next time.

3.2.2.7 INT WINAPI RF_M1_INITVAL

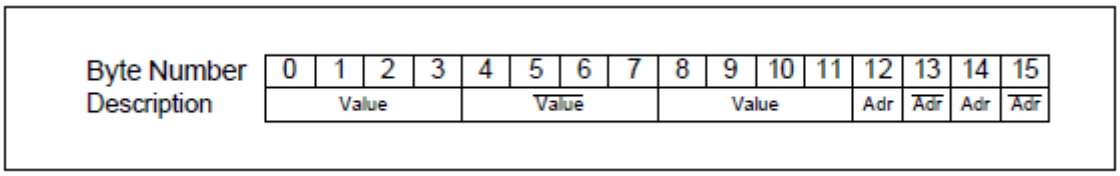
Function: Mifare_Std card Initialize Value

Prototype: int WINAPI rf_M1_initval (unsigned short icdev,
unsigned char block,
long value)

Parameter: icdev: [IN] Device ID
block: [IN] block absolute address
pValue: [IN] initialize purse value at HEX format, low byte in former, for example, decimal 100, change to hexdecimal is 64, but it needs 4 bytes, so the data is 64000000, it store in the block's sequence is b0b1b2b3.

Return: return 0 if successful

Note: to use a block as a value block, this block needs to initialize into value format.



3.2.2.8 INT WINAPI RF_M1_READVAL

Function: Mifare_Std Read Value
Prototype: int WINAPI rf_M1_readval (unsigned short icdev,
unsigned char block,
long *pValue)
Parameter: icdev: [IN] Device ID
block: [IN] block absolute address
pValue: [OUT] response value at HEX format, low byte in former
Return: return 0 if successful

3.2.2.9 INT WINAPI RF_M1_INCREMENT

Function: Mifare purse increment
Prototype: int WINAPI rf_M1_increment (unsigned short icdev,
unsigned char block,
long value)
Parameter: icdev: [IN] Device ID
block: [IN] block absolute address
value: [IN] increase value at HEX format, low byte in former
Return: return 0 if successful

3.2.2.10 INT WINAPI RF_M1_DECREMENT

Function: Mifare purse decrement
Prototype: int WINAPI rf_M1_decrement (unsigned short icdev,
unsigned char block,
long value)
Parameter: icdev: [IN] Device ID
block: [IN] block absolute address
value: [IN] decrease value at HEX format, low byte in former
Return: return 0 if successful

3.2.2.13 INT WINAPI RF_HALT

Function: Mifare Halt
Prototype: int WINAPI rf_halt (unsigned short icdev)
Parameter: icdev: [IN] Device ID
Return: return 0 if successful
Note: if the card was halt, then it will not respond. To active this card, it has to move away the

reader and move into again or you can send command to switch the antenna to active the card.

4. ERROR CODE

Error Code	Meaning
1	Baud rate error
2	Port error or Disconnect
10	General error
11	undefined
12	Command Parameter error
13	No card
20	Request failure
21	Reset failure
22	Authenticate failure
23	Read block failure
24	Write block failure
25	Write address failure
26	Write address failure

---

5. Serial Protocol

If you need to development your own programs, you can use this protocol.

5.1. Communication Setting

The communication protocol is byte oriented. Both sending and receiving bytes are in hexadecimal format. The communication parameters are as follows,

Baud rate: 115200 bps (default)
Data: 8 bits
Stop: 1 bit
Parity: None
Flow control: None

5.2. Command Format

Data format		Binary HEX “hexadecimal”				
Data package						
Head	Length	Node ID	Function Code	Data …	XOR	

SEND DATA FORMAT:

	Data length (Byte)		X O R	S U M
Head	02	Fixed: 0xAA , 0xBB		
Length	02	There are several effective bytes that including XOR follows this column.	FF	00
Node ID	02	Destination Node Address Number. xx xx: Low byte first 00 00: Broadcast to each reader.	X	S
Function code	02	It will be transmission ability of each different command. Low byte frist	X	S
Data	00~D0	Data length is not fixed, according to its purpose.	X	S
XOR	01	XOR each byte from Node ID to Last Data byte with 0xFF.		S

RESPOND DATA FORMAT:

	Data length (Byte)		X O R	S U M
Head	02	Fixed: 0xAA, 0xBB		
Length	02	There are several effective bytes that including XOR follows this column.	FF	00
Node ID	02	Destination Node Address Number. xx xx: Low byte first 00 00: Broadcast to each reader.	X	S
Function code	02	It will be transmission ability of each different command. Low byte frist	X	S
Status	1	Reply result, if succeed is 0, other fail.		
Data	00~D0	Data length is not fixed, according to its purpose.	X	S
XOR	01	XOR each byte from Node ID to Last Data byte		S

NOTE: If from "Length" to "XOR" have a data is "AA" then should follow a data "0x00", but length don't changed.
While a command send and after 100ms no reply then consider this command failed.

5.3.0 COMMAND LIST

No.	Meaning	Code
1	Initialize port	0x0101
2	Set device node number	0x0102
3	Read device node number	0x0103
4	Read device Mode	0x0104
5	Set buzzer beep	0x0106
6	Set Led color	0x0107

7	**RFU**	**0x0108**
8	**Set antenna status**	**0x010c**
9	**Mifare Request**	**0x0201**
10	**Mifare anticollision**	**0x0202**
11	**Mifare Select**	**0x0203**
12	**Mifare Hlta**	**0x0204**
13	**Mifare Authentication2**	**0x0207**
14	**Mifare Read**	**0x0208**
15	**Mifare Write**	**0x0209**
16	**Mifare Initval**	**0x020A**
17	**Mifare Read Balance**	**0x020B**
18	**Mifare Decrement**	**0x020C**
19	**Mifare Increment**	**0x020D**

5.3.1. Initialize port：0x0101

Function：Set baud rate

Format： AA BB 06 00 00 01 01 “Baud_parameter” “xor Chk”

Baud_parameter：

0 = 4800;
1 = 9600;
2 = 14400;
3 = 19200;
4 = 28800;
5 = 38400;
6 = 57600;
7 = 115200;

Host Send to Reader Example：

Send： AA BB 06 00 00 00 01 01 03 03 //Set Baud Rate as 19200

Respond：AA BB 06 00 bf ff 01 01 00 40

5.3.2. Set device node number：0x0102

Host Send to Reader Example：

Send： AA BB 07 00 00 00 02 01 00 00 03 //Set device node number = 0x00 00

5.3.3. Read device node number：0x0103

Host Send to Reader Example：

Send： AA BB 05 00 00 00 03 01 02 //Read device node number

5.3.4. Read device Mode：0x0104

Function：Read device mode and version

Host Send to Reader Example：

Send： AA BB 05 00 00 00 04 01 05

Respond：AA BB 12 00 52 51 04 01 00 *59 48 59 36 33 32 41 2D 31 32 30 33* 11

"*59 48 59 36 33 32 41 2D 31 32 30 33*" is "YHY632A-1203"

5.3.5. Set buzzer beep：0x0106

Function：Beep

Format： AA BB 06 00 00 00 06 01 Delay XOR

Delay*10ms beep time, XOR is xor check

Host Send to Reader Example：

Send： AA BB 0600 00 0006 01 6463

Respond：AA BB060052 5106010004

5.3.6. Set Led color：0x0107

Host Send to Reader Example：

Send： AA BB 06 00 00 00 07 01 03 05 //Set Red&green LED on

Respond：AA BB 06 00 bf bf 07 01 00 06

Tenth data is LED parameter, function as below:

00 = LED_RED Off, LED_BLUE Off

01 = LED_BLUE On, LED_RED Off

02 = LED_BLUE Off, LED_RED On

5.3.7. Reader working status：0x0108，not use in this device

5.3.8. Antenna status：0x010c

Host Send to Reader Example：

Send： AA BB 06 00 00 00 0c 01 *00* 0D //Set antenna off 。

"*00*" is Antenna status parameter:

00 = Close Filed, 01= Open Filed

5.3.9. Mifare Request：0x0201

Function：Request Type a Card

Format： AA BB 06 00 00 00 01 02 req_code XOR

req_code: Request mode:

req_code: 0x52: request all Type A card In filed

req_code: 0x26: request idle card

Host Send to Reader Example：

Send： AA BB 06 00 000001 0252 51

Respond：AA BB 0800 52 51 01 02 00 *04 00* 04

TagType： 0x4400 = ultra_light

0x0400 = Mifare_One(S50)

0x0200 = Mifare_One(S70)

0x4403 = Mifare_DESFire
0x0800 = Mifare_Pro
0x0403 = Mifare_ProX

5.3.10. Mifare anticollision：0x0202

Function：Card anticollision

Format： AA BB 05 00 00 00 02 02 00

Respond：AA BB 0a0052 51 02 02 00 *46 ff a6 b8* a4

"*46 ff a6b8*" is card serial number

5.3.11. Mifare Select：0x0203

Function：Select card

Format： AA BB 09 00 00 00 03 02 xx xx xx xx XOR

Ninth to twelfth is card serial number 。

Host Send to Reader Example：

Send： AA BB 09 00 00 00 03 02 46 ff a6 b8 a6

Respond：AA BB 07 00 52 51 03 02 00 08 0a

5.3.12. Mifare Hlta：0x0204

Function：Hlta card

Host Send to Reader Example：

Send： AA BB 05 00 0000 04 02 06

Respond：AA BB 06 00 52 51 04 02 00 05

5.3.13. Mifare Authentication2：0x0207

Function：Authenticate Card

Format： AA BB xx 00 00 00 07 02 Auth_mode Block xx xx xx xx xx xx XOR

Auth_mode:	Authenticate mode, 0x60: KEY A, 0x61: KEY B
Block:	Authenticate block

Host Send to Reader Example：

Send： AA BB 0d 00 00 00 07 02 60 04 ff ff ff ff ff ff 61

Authenticate Block 4, Key A = "FF FF FF FF FF FF"

Respond：AA BB 0600 52 51 07 02 00 06

5.3.14. Mifare Read：0x0208

Function：Read block

Format： AA BB 06 00 00 00 08 02Block XOR

Block = which block want read

Host Send to Reader Example：

Send： AA BB 06 00 00 0008 02 040e

Respond：AA BB 16 00 52 51 08 02 00 00 00 00 00 00 00 00 00 00 00 00 00 12 34 56 78 01

Tenth to sixteenth byte is Data

5.3.15．Mifare Write：0x0209

Function：Write block

Format： AA BB 16 00 00 00 0902 Block D0 D1 D2 D3 D4 D5 D6D7 D8 D9 Da Db Dc Dd De Df XOR

Sample：Write data to Block4

Host Send to Reader Example：

Send： AA BB 16 00 00 00 09 02 04 00 00 00 00 00 00 00 00 00 00 00 00 12 34 78 56 07

Respond：AA BB 06 00 52 51 09 02 00 08

5.3.16．Mifare Initval：0x020A

Function：Initialize purse

Format： AA BB 0a 00 00 00 0a 02 Block V0V1V2V3 XOR

5.3.17．Mifare Read Balance：0x020B

Function：Read balance

Format： AA BB 06 00 00 00 0B 02 Block XOR Return four byte balance

5.3.18．Mifare Decrement：0x020C

Function：Decrease balance

Format： AA BB 0a 00 00 00 0c 02 Block V0V1V2V3 XOR

5.3.19．Mifare Increment：0x020D

Function：Increase balance

Format： AA BB 0a 00 00 00 0D02 Block V0V1V2V3 XOR
